Approximations on Minimum Weight Pseudo-Triangulation problem using Ant Colony Optimization Metaheuristic

Edilma Olinda Gagliardi
Maria Gisela Dorzán
Mario Guillermo Leguizamón
Facultad de Ciencias Físico Matemáticas y Naturales
Universidad Nacional de San Luis
San Luis, Argentina
{*oli, mgdorzan, legui* }*@unsl.edu.ar*

Gregorio Hernández Peñalver
Facultad de Informática
Universidad Politécnica de Madrid
Madrid, España
gregorio@fi.upm.es

***Abstract*—In this work, we consider the Minimum Weight Pseudo-Triangulation (MWPT) problem of a given set of n points in the plane. Globally optimal pseudo-triangulations with respect to the $weight$, as optimization criteria, are difficult to be found by deterministic methods, since no polynomial algorithm is known. We show how the Ant Colony Optimization (ACO) metaheuristic can be used to find high quality pseudo-triangulations of minimum weight. We present the experimental and statistical study based on our own set of instances since no reference to benchmarks for these problems were found in the literature. Throughout the experimental evaluation, we appraise the ACO metaheuristic performance for MWPT problem.**

***Keywords*-Pseudo-Triangulation, Minimum Weight, Computational Geometry, ACO Metaheuristic.**

I. Introduction

Pseudo-triangulations are planar partitions that appear as data structures in Computational Geometry, as planar bar-and-joint frameworks in rigidity theory and as projections of locally convex surfaces. They have arisen in the last decade as interesting geometric combinatorial objects with connections and applications in visibility, rigidity theory and motion planning.

There are many possible optimality criteria, often based on edge lengths, angles, or area. So, in Computational Geometry there are many optimization problems that either are NP-hard or no polynomial algorithms are known to solve them. Optimization problems related to special geometric configurations such as pseudo-triangulations are interesting to research due to their use in many fields of application.

A pseudo-triangle is a simple polygon with three convex vertices, and a pseudo-triangulation is a tiling of a planar region into pseudo-triangles. The weight of a pseudo-triangulation is the total Euclidean edge length. Computing a minimum weight pseudo-triangulation for a point set in the plane has been one of the main optimality criteria for minimizing the total length for pseudo-triangulations. Indeed, the Minimum Weight Pseudo-Triangulation (MWPT) problem minimizes the sum of the edge lengths, providing a quality measure for determining how good structure is.

Find globally optimal pseudo-triangulations with respect to the $weight$, are difficult to be found by deterministic methods, since no polynomial algorithm is known. The complexity of MWPT problem is unknown, but Levcopoulos and Gudmundsson [8] show that a 12-approximation of an minimum weight triangulation can be computed in $O(n^3)$ time. They give an $O(\log n \cdot w(MST))$ approximation of an minimum weight triangulation, in $O(n \log n)$ time, where $w(MST)$ is the weight of the minimum Euclidean spanning tree, which is a subset of the obtained structure.

The approximate algorithms arise as alternative candidates for MWPT problem. These algorithms can obtain approximate solutions to the optimal ones.

The metaheuristic methods can be specific for a particular problem or they can be part of a general applicable strategy in the resolution of different problems. These iterative generation processes guide the search of solutions intelligently combining different concepts of diverse fields as artificial intelligence [11], biological evolution [2], swarm intelligence [9], among others.

They have a simple implementation and they can efficiently find good solutions for NP-hard optimization problems [10]. For the experimental study presented in this work we use the *Ant Colony Optimization* (ACO) metaheuristic.

Previous works about approximations on MWPT problem using metaheuristic, were presented in [4] and [7], where we described the design of the ACO algorithms and gave the first steps in this research. According to the current state-of-the-art about the MWPT problem, we continue the research considering the metaheuristic techniques as the more appropriate approach to find near optimal solutions.

This paper is organized as follows. In the next Section, we present the theoretical aspects of MWPT problem.

Subventions Researh Project TABD-N22/F014-UNSL, CONICET Agentina, and Proyect MTM2008-05043-Ministerio de Ciencia e Innovacion-España.

Following, we describe the general overview of the ACO metaheuristic and the proposed ACO algorithm for the MWPT problem, namely ACO-MWPT. Next Section, we present the experimental and statistical study. Last Section is reserved for the conclusions and future vision.

II. Minimum Weight Pseudo-Triangulation

Let *S* be a set of points in the plane. A pseudo-triangulation *PT* of *S* is a partition of the convex hull of *S* into pseudo-triangles whose set of vertices is exactly *S*. A pseudo-triangle is a planar polygon that has exactly three convex vertices, called *corners* (see Figure 1). The weight of a pseudo-triangulation *PT* is the sum of the Euclidean lengths of all the edges of *PT*.

The pseudo-triangulation that minimizes this sum is named a *Minimum Weight Pseudo-Triangulation* of *S* and it is denoted by $MWPT(S)$.

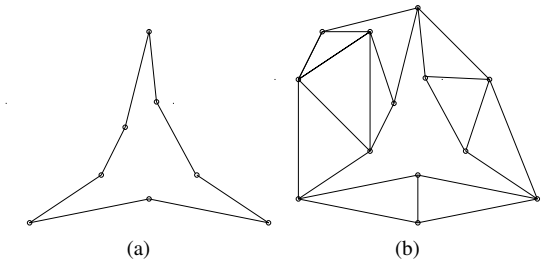

Figure 1: Examples of (a) a pseudo-triangle and (b) pseudo-triangulation.

The concept of pseudo-triangulation was introduced by Pocchiola and Vegter in [12] on the analogy of the arrangements of pseudo-lines; see [13] for a survey with many results of pseudo-triangulations.

As we mentioned in Section 1, there exists a set of points for which any triangulation will have weight $O(n \cdot wt(M(S)))$. A natural question is whether there exist a similar worst-case bounds for pseudo-triangulations. Rote et al. [14] were those who asked if the MWPT is a NP-hard problem, stimulating the search of exact or approximate algorithms.

Gudmundsson and Levcopoulos [8] considered the problem of computing a minimum weight pseudo-triangulation of a set S of n points in the plane, presenting an $O(n \cdot logn)$-time algorithm that produces a pseudo-triangulation of weight $O(logn.wt(M(S)))$ which is shown to be asymptotically worst-case optimal. That is, there exists a point set S for which every pseudo-triangulation has weight $\Omega(logn.wt(M(S)))$, where $wt(M(S))$ is the weight of a minimum spanning tree of S. Also, they presented a constant factor approximation algorithm running in cubic time, and they gave an algorithm that produces a minimum weight pseudo-triangulation of a simple polygon.

It is also worth noticing that to the best knowledge of the authors, there are no approaches using metaheuristic techniques for solving MWPT problem.

III. Ant Colony Optimization Metaheuristic

The ACO metaheuristic involves a family of algorithms in which a colony of artificial ants cooperate in finding good solutions to difficult discrete optimization problems. Cooperation is a key design component of ACO algorithms. The idea is to allocate the computational resources to a set of relatively simple agents (artificial ants) that communicate indirectly by stigmergy. Thus, good quality solutions are an emergent property of the agents cooperative interaction. An artificial ant in an ACO algorithm is a stochastic constructive procedure that incrementally builds a solution by adding opportunely defined solution components to a partial solution under construction. Therefore, the ACO metaheuristic can be applied to any combinatorial optimization problem for which a constructive graph can be defined. Each edge (i, j) in the graph represents a posible path and it has associated two information sources that guide the ant moves: pheromone trails and heuristic information. The pheromone trail, denoted by τ_{ij}, encodes a long-term memory about the entire ant search process, and is updated by the ants themselves. The heuristic information, denoted by η_{ij}, represents *a priori* information about the problem instance or run-time information provided by a source different from the ants. In many cases η is the cost, or an estimate of the cost, of adding the component or connection to the solution under construction.

These values are used by the ants to make probabilistic decisions on how to move on the graph. The ants act concurrently and independently and although each ant is complex enough to find a solution to the problem, which is probably poor, good-quality solutions can only emerge as the result of the collective interaction among the ants. This is obtained via indirect communication mediated by the information that ants read or write in the variables storing pheromone trail values. It is a distributed learning process in which the single agents, the ants, are not adaptive themselves but, on the contrary, adaptively modify the way the problem is represented and perceived by other ants [3].

There are two additional process for updating pheromone and the daemon actions. The pheromone updating is the process by which the pheromone trails are modified. The trail values can either increase, as ants deposit pheromone on the components or connections they use, or decrease, due to pheromone evaporation. The daemon procedure is used to implement centralized actions which cannot be performed by single ants. Examples of daemon actions are the activation of a local optimization procedure, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a nonlocal perspective. The daemon can observe the path found by each ant in the colony

and select one or a few ants, like those that built the best solutions in the algorithm iteration that allowed to deposit additional pheromone on the connections they used.

We present an ACO algorithm called MWPT-ACO, the description of its main components for MWPT problem and we describe in detail the specific *BuildSolutionk* function.

Algorithm MWPT-ACO

Initialize
for $c \in \{1, \ldots, C\}$ **do**
  for $k \in \{1, \ldots, K\}$ **do**
    BuildSolutionk
    EvaluateSolution
  end for
  SaveBestSolutionSoFar
  UpdateTrails
end for
ReturnBestSolution

Main components of MWPT-ACO algorithm:

- *Initialize*: The initial trail of pheromone associated to each edge is τ_0; it is an small positive value, in general, the same for all edges. The quantity of ants of the colony is *K*. The weights define the proportion in which they will affect the heuristic information and pheromone trails in the probabilistic transition rule, named respectively β and α. The maximum number of cycles is *C*.
- *BuildSolutionk*: this process begins with a empty solution which is extended at each step by adding a feasible solution component chosen from the current solution neighbors; i.e., to find a route on the construction graph guided by the mechanism that defines the set of feasible neighbors with regard to the partial solution. The choice of a feasible neighbor is done in a probabilistic way in every step of the construction, depending on the used ACO variant. In this work, the selection rule for the solutions construction is based on the following probabilistic model:

$$P_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{h \in F(i)} \tau_{ih}^{\alpha} \cdot \eta_{ih}^{\beta}}, & j \in F(i); \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

  - $F(i)$ is the set of feasible points for point i.
  - τ_{ij} is the pheromone value associated to edge (i, j).
  - η_{ij} is the heuristic value associated to edge (i, j).
  - α and β are positives parameters for determining the relative importance of the pheromone with respect to the heuristic information.
- *EvaluateSolution*: evaluates and saves the best solution found by ant k in the current cycle.
- *SaveBestSolutionSoFar*: saves the best solution found for all cycles so far.
- *UpdateTrails*: increases the pheromone level in the promising paths, and is decreased in other case. First, all the pheromone values are decreased by means of the process of evaporation. Then, the pheromone level is increased when good solutions appear. The following equation is used:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \Delta\tau_{ij} \quad (2)$$

  - $\rho \in (0, 1]$ is the factor of persistence of the trail.
  - $\Delta\tau_{ij} = \sum_{k=1}^{K} \Delta^k \tau_{ij}$ is the accumulation of trail, proportional to the quality of the solutions.
  - $\Delta^k \tau_{ij} = \begin{cases} Q/L_k, & \text{when ant } k \text{ used edge } (i, j); \\ 0, & \text{in other case.} \end{cases}$
  - *Q* is a constant depending of the problem; it usually set to 1.
  - L_k is the objective value of the solution *k*.

  Pheromone evaporation avoids a fast convergence of the algorithm. In addition, this way of forgetting allows the exploration of new areas of the search space. The update of the pheromone trail can be done according to one of the following criteria: *elitist* and *not elitist*. In the elitist case, the best found solution is used to give an additional reinforcement to the levels of pheromone. The not elitist one uses the solutions found by all the ants to give an additional reinforcement to the levels of pheromone.

For *BuildSolutionk* process, each ant builds a pseudo-triangulation, starting with one face. This face has the edges obtained by the convex hull of the points set S, i.e., $CH(S)$. For the solution construction, each ant performs a process of partitioning set S in faces. This process finishes when all faces are pseudo-triangles without interior points. A face is divided into two faces when it has interior points or is not a pseudo-triangle. Thus, the partition can be done if $i)$ there are at least one interior point and two points in the border; or $ii)$ there is not an interior point, so we use two points located on the border. $Faces_k$ represents the set of no treated faces.

Algorithm BuildSolutionk

$S_k \leftarrow \emptyset$
while $(Faces_k \neq \emptyset)$ **do**
  Let *F* be a face in $Faces_k$
  if *F* is Pseudo-triangle without interior points **then**
    $S_k \leftarrow S_k \cup \{F\}$ /* *F* is a new pseudo-triangle */
    $Faces_k \leftarrow Faces_k - \{F\}$
  else
    $PartitionFace(F)$
  end if
end while

PartitionFace(*F*) selects the points of *F* to build the new faces. It takes one interior point and two probabilistic selected points of the border, or (if there is not interior point) only two probabilistic selected points of the border. The feasible points for a point are those visible and not adjacent to it. The selection is done according to Equation 1. Also, this process uses two additional procedures *SelectInteriorPoint*(*F*) and *SelectBorderPoint*(*F*), where the point is randomly selected.

IV. Experimental Evaluation and Statistical Analysis

In this work we present an ACO algorithm for the MWPT problem. The proposed ACO algorithm are represented by an Ant System (AS), a particular instance of the class of ACO algorithms.

We will refer to the following aspects: experimental evaluation, the statiscal analysis and the computational effort of the algorithms.

With respect to the experimental phase, we have obtained evaluations that guide the experimentation, and it allows us to decide which are most suitable parameter settings. We try to find an acceptable combination of parameter values for the ACO-MWPT algorithm in order to obtain pseudo-triangulations of small weight, closer to the minimum.

The collections of instances for MWPT problem were designed by the authors, using an *instance generator*. The instance generator uses different functions of CGAL Library [1]. This generator was done since to the best knowledge of the authors there not exist in the literature benchmarking data publicly available that allow us to compare our proposal with other algorithms. A collection of 10 instances of size 40/80/120/160/200 were generated; i.e., a total of 50 problem instances, each one is called LDn-i, the size of the i-instance, $1 \leq i \leq 10$, is denoted by n. The points are randomly generated, uniformly distributed with coordinates $x, y \in [0, 1000]$. There are no collinear points.

We present representative cases for the experimental research done, considering four instances LD40-1, LD40-2, LD40-3, LD40-4, LD80-1, LD80-2, LD80-3 LD80-4, LD120-1, LD120-2, LD120-3, and LD120-4, of size 40, 80, and 120 respectively.

The ACO-MWPT and auxiliary algorithms were implemented in C language. The software was run on BACO parallel cluster, under CONDOR batch queuing system. Considering different combination of parameter values, the current experimental study is devoted to analyze the performance of the algorithms with respect the quality of the solutions.

We used the following parameter values: $\alpha = 1$; $\beta = 1$ and 5; and $\rho = 0.10$, 0.25, and 0.50. *elit* = 1 and 0 (1 for trails updated in a elitist way; in other case, the updating is done in a not elitist way).

The number of cycles, *C*, is 1000; the number of ants, *K*, is 50. For each parameter setting, 30 runs were performed by using different random seeds. For each instance, the experiment were done with the twelve parameter settings, according to combinations of parameters before detailed. We considered the objective function (*weight*) for obtaining average, median, best, and standard deviation values, and also the pseudo-triangles number.

We show the results according to the four best parameter settings considering the four *Best* values. We only show the results for the best four parameter settings since the results for the remaining ones were of lower quality with respect to the best found values.

Next, we show the experimental results, considering the above presented settings. Each parameter setting is denoted by ($instance$-β-ρ-$elit$); α is ever equal 1. The decimal numbers are not showed because they are not significant.

We analyze the performance of the ACO-MWPT algorithm over instances of 40, 80, and 120 points. The results for this experimental study are resumed in Tables V to VII (see Appendix).

We show the results according to the four best parameter settings with respect to the smaller weights. There is not a unique parameter combination that obtains the best results. In addition, we show an extra data #Pts, which represents quantity of pseudo-triangles in a pseudo-triangulation. Note that the pseudo-triangulations with lower weight values do not have fewer pseudo-triangles. This observation is interesting because intuitively it is possible to think that the pseudo-triangulations of the lowest weight have fewer pseudo-triangles, or they are minimum weight triangulations, but the opposite was demonstrated by experimentation.

Furthermore, we considered necessary to observe globally the influence of parameters. Table I is a summary of the previous tables and shows that the best weights are obtained using configurations with $\beta = 5$, $elit = 0$, and 1, and $\rho = 0.1, 0.25$, and 0.5, i.e., we obtained better results giving more relevance to the heuristic information.

Table I: ACO-MWPT: Summary of results for four instances of 40, 80, and 120 points with respect to the best values.

β	ρ	$elit$
1 (6.25%)	0.10 (39.58%)	0 (50%)
5 (93.75%)	0.25 (35.42%)	1 (50%)
	0.50 (25%)	

To better assess our proposal, we compare the ACO-MWPT algorithm with a greedy algorithm called Greedy Pseudo-Triangulation (GPT). The greedy strategy was mainly chosen for two reasons: a) there not exist any greedy strategy for MWPT problem and b) this strategy is similar to the ACO metaheuristic since both approaches perform a constructive process for generating a pseudo-triangulation. Nevertheless, they apply a different criteria at the component selection procedure.

The GPT algorithm builds a single pseudo-triangulation, starting with one face. The initial face has the edges in the convex hull. At construction step, it performs a process of partitioning the set S in new faces. This process finishes when all faces are pseudo-triangles, which have not interior point. A face is divided into two faces if it has some interior point, or is not a pseudo-triangle. Thus, the partition can be done, by selecting: *i)* one interior point p and two points, q and r, that they are in the face border, where q and r are the closest to p, or *ii)* when there is not an interior point, two points on the border are selected, which are closest each other.

Table II shows the lower weights found and the respective number of pseudo-triangles (#Pts). It can be seen there not exist a clear correlation between the objective values and #Pts, i.e., a lower value of the objective function do not imply a lower value of #Pts for the respective instance.

Table II: ACO-MWPT: Comparing results between ACO-MWPT and Greedy strategy.

Instance	ACO-MWPT	# Pts ACO	GPT-MWPT	# Pts GPT
LD40-1	6115636	51	5312131	56
LD40-2	4442710	49	4292347	52
LD40-3	5684342	49	5794018	58
LD40-4	5627098	48	6245196	57
LD80-1	7898497	105	7458787	113
LD80-2	9584718	104	8931272	106
LD80-3	8918853	106	6516103	107
LD80-4	8004652	110	7393297	112
LD120-1	12842149	163	14097967	163
LD120-2	9247582	154	7106543	174
LD120-3	12326883	167	11519206	160
LD120-4	10647886	170	8341281	175

In order to statistically analyze the effect of each parameter on the behavior of the ACO-MWPT algorithm, we studied by considering different parameter settings. The setting were empirically derived after numerous experiments. We get 12 parameter settings for each environments for each instance, they are listed and identified in Table III. The values for α and $criterion$ are always one ($\alpha = 1$ and $criterion = 1$).

We performed the Kolmogorov-Smirnov test to show the sample does not follow a normal distribution. Therefore we use a non-parametric statistical test to evaluate the algorithms.

The analysis we perform has two phases. First, we apply the Kruskal-Wallis test to perform the median comparison in order to determine the sensitivity of the parameters, using the parameter settings given in Table III. The null hypothesis considered is: there is not a significative difference among the found results and if there are differences, they are due to random effects.

Table III: Parameter settings and their identifiers (ID).

ID	β	ρ	$elit$
1	1	0.10	0
2	1	0.25	0
3	1	0.50	0
4	5	0.10	0
5	5	0.25	0
6	5	0.50	0
7	1	0.10	1
8	1	0.25	1
9	1	0.50	1
10	5	0.10	1
11	5	0.25	1
12	5	0.50	1

Then we apply the Tukey method in order to determine the experimental conditions where exist significative differences.

On the second phase of the statistical analysis, we carried out the boxplot method to visualize the distribution of the weights for each environment.

We show the respective statistical analysis for the ACO-MWPT algorithm over the same group of instances. The $y\text{-}axis$ represents the identifier (ID) for each parameter setting shown in Table III. In this case, the parameter settings 1, 2, and 3 show, in most of the cases, have significative differences with respect to the remaining ones for all instances of size 40 (Figure 2). This could means that a variation of parameters $\beta = 5$ and $elit = 0, 1$; or $\beta = 1$ and $elit = 1$ will no produce significative differences in the results. However, for the instances of size 80 and 120 (Figures 3 and 4) the parameter settings 1, 2, 3, 7, 8, and 9 are those which produce (not always) a more similar behavior, as well as the parameter setting 4, 5, 6, 10, 11, and 12; on the other side.

Unfortunately, there is no clear behavioral pattern for these instances. Nevertheless, it can be said that the use elitism has not clear influence in the final results. Interesting, the same observation given before can be applied to the quality and dispersion of obtained results as seen in the boxplots displayed in Figures 5, 6, and 7. However, from the point of view of the solution quality, ACO-MWPT algorithm obtained the worst results by using the parameter settings 1, 2 and 3 for the instances of size 40 (see Figure 5). The $x\text{-}axis$ represents the identifier (ID) for each parameter setting shown in Table III and the $y\text{-}axis$ represents the weight. Similarly, for instances of size 80 and 120 (see Figures 6 and 7) the best performance (either in terms of median and best values) is achieved under the parameter settings 4, 5, 6, 10, 11, and 12. This means that with $\alpha = 1$ and $\beta = 5$, ACO-MWPT algorithm achieves the best performance independently of parameters ρ and $elit$.

With respect to the runtimes analysis, we compare and analyze the computational effort of the algorithms applied to the MWPT problem (i.e., ACO-MWPT and GPT). The ACO-MWPT algorithm is iterative and stochastic

population-based algorithms; while the GPT algorithm is a deterministic algorithm. So, it builds only one solution on a time bounded by almost $O(n^3)$ for GPT. Certainly, this will show important differences when comparing the respective runtimes of these algorithms. Although ACO algorithms consume more computational resources (mainly time) when compared respectively to the two deterministic algorithms, they found higher quality solutions. In addition, it is worth remarking that there are some applications in Computational Geometry related to the proposed problems which require high quality solutions. Consequently, more complex and time-consumer algorithms need to be used to reach solutions of higher quality as usually expected. Nevertheless, we are aware that for some Computational Geometry applications other methods could be a simple and direct alternative when solutions of medium or low quality are acceptable.

In Table IV we show the runtimes of ACO-MWPT algorithm considering the parameter settings that yields the best results. In regards of the ACO algorithm, the α, β and ρ parameters do not substantially affect the runtime because they only modify the probability distribution and the extent of amount of pheromone laid on the edges. Therefore, as the consumed time does not significantly vary for different values of α, β, and ρ we particularly show the runtimes for $\alpha = 1$, $\beta = 1$, $\rho = 0.10$, and $elit = 0$ using one instance of each one of the three different problem sizes studied here.

Table IV: MWPT: Average runtimes for ACO-MWPT algorithm (for 30 seeds) and the runtime for Greedy Pseudo-Triangulation (in milliseconds).

# points	ACO-MWPT	GPT
40	120431	69
80	328451	83
120	478487	94

As previously analyzed and expected, Table IV show the runtimes of the algorithms applied. It can clearly be observed the increment of the computational cost of the ACO algorithm with respect to GPT. These results show that solutions of higher quality can be found by applying a metaheuristic technique but a higher cost. This situation encourages us to go further in our research in order to design either an improved version of the ACO algorithm studied here and also, to propose alternative metaheuristics that reduce the computation time required to reach high quality solutions (the authors are currently working in an improved version of Simulated Annealing to deal with the problems presented here).

V. Conclusion

Considering MWPT problem we proposed ACO algorithm for obtaining approximations on minimum weight pseudo-triangulations.

The proposed ACO algorithm is represented by an Ant System (AS), a particular instance of the class of ACO algorithms.

We also detailed the generation of instances for the experimental evaluation, being this another contribution of this paper, since there are not available instances with special properties for building pseudo-triangulations. At the moment, we have a large collection of instances for future experimentations, where the obtained results could be used as benchmarks.

Accordingly, future work will address the use of improved parameter setting for the ACO algorithms and the experimentation with the whole collection of instances generated. In addition, we aim at comparing the proposed algorithms against other metaheuristics. Currently we are conducting an experimental study involving Simulated Annealing [5] [6] which will help us to compare the performance of both techniques in regards of quality of solutions and runtimes.

Acknowledgment

The authors would like to thank to Research Project Tecnologías Avanzadas de Bases de Datos 22/F014 financed by Universidad Nacional de San Luis, San Luis, Argentina; CONICET-AGENTINA; Instituto de Física Aplicada (INFAP)-UNSL-CONICET to allow us to use the cluster; and Research Project MTM2008-05043 del Ministerio de Ciencia e Innovacion-España. Likewise, to the colleagues who contributed with opinions and technical advice.

References

[1] Cgal, computational geometry algorithms library.

[2] T. Bäck, D. Fogel, and Z. Michalewicz. *Handbook of evolutionary computation*. Oxford Univ. Press, 1997.

[3] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.

[4] M. Dorzán, E. Gagliardi, Leguizamón, and G. M., Hernández Peñalver. Algoritmo aco aplicado a la obtención aproximada de triangulaciones de peso mínimo. In *XXXV Conferencia Latinoamericana de Informática*, 2009.

[5] M. Dorzán, E. Gagliardi, M. Leguizamón, and G. Hernández Peñalver. Simulated annealing aplicado a triangulaciones y pseudotriangulaciones de peso mínimo. In *XVI Congreso Argentino de Ciencias de la Computación*, 2010.

[6] M. Dorzán, E. Gagliardi, M. Leguizamón, and G. Hernández Peñalver. Triangulaciones y pseudotriangulaciones de peso mínimo: resolución aproximada con simulated annealing. In *VII Jornadas de Matemática Discreta y Algorítmica. España*, 2010.

[7] M. Dorzán, E. Gagliardi, M. Leguizamón, M. Taranilla, and G. Hernández Peñalver. Algoritmos aco aplicados a problemas geométricos de optimización. In *XIII Encuentros de Geometría Computacional*, 2009.

[8] J. Gudmundsson and C. Levcopoulos. Minimum weight pseudo-triangulations. *Comput. Geom.*, 38(3):139–153, 2007.

[9] J. Kennedy and R. Eberhart. *Swarm Intelligence (The Morgan Kaufmann Series in Artificial Intelligence)*. Morgan Kaufmann, 1st edition, April 2001.

[10] Z. Michalewicz and D. Fogel. *How to Solve It: Modern Heuristics*. Springer, 2004.

[11] I. Osman and J. Kelly. *Meta-heuristics: Theory and Applications*. Kluwer academic publishers, 1996.

[12] M. Pocchiola and G. Vegter. Pseudo-triangulations: Theory and applications. In *Symposium on Computational Geometry*, pages 291–300, 1996.

[13] G. Rote, F. Santos, and I. Streinu. *Pseudo-triangulations — a survey*. Contemporary Mathematics. American Mathematical Society, December 2008.

[14] G. Rote, C. Wang, L. Wang, and Y Xu. On constrained minimum pseudotriangulations. In Proc. 9th Intern. Comp. Comb. Conf., 2003.

APPENDIX

Table V: MWPT: Results for four instances of 40 points.

Par. Setting	Average	Median	Best	Std. Dev.	#Pts
LD401-5-0.10-1	6557443	6607908	6115636	166770	51
LD401-5-0.25-1	6644026	6658654	6286985	166104	48
LD401-5-0.50-1	6669542	6713656	6320652	159069	49
LD401-5-0.50-0	6777518	6847951	6322956	158225	52
LD402-5-0.25-1	4748353	4757694	4442710	114885	49
LD402-5-0.10-0	4681136	4685804	4470550	69468	48
LD402-5-0.10-1	4707699	4747199	4490214	83905	50
LD402-5-0.25-0	4729018	4749318	4524206	77542	43
LD403-5-0.25-1	6069210	6071705	5684342	143063	49
LD403-5-0.10-1	5980440	6021063	5699513	136174	50
LD403-5-0.25-0	6075029	6118394	5744775	110439	45
LD403-5-0.50-0	6073308	6104511	5746463	121285	51
LD404-1-0.50-1	6236883	6258985	5627098	218860	48
LD404-5-0.10-1	6162888	6154961	5668910	166455	49
LD404-5-0.50-1	6229822	6237045	5869145	202030	50
LD404-5-0.50-0	6237883	6252135	5903381	139767	47

Table VI: MWPT: Results for four instances of 80 points.

Par. Setting	Average	Median	Best	Std. Dev.	#Pts
LD801-5-0.50-0	8281137	8300956	7898497	160360	105
LD801-5-0.10-0	8325983	8331038	7923788	149888	109
LD801-5-0.25-0	8304994	8339204	7928177	163459	109
LD801-5-0.10-1	8332003	8363583	7988963	127377	105
LD802-5-0.25-1	10512726	10604489	9584718	362414	104
LD802-5-0.25-0	10427016	10476297	9673011	270506	111
LD802-5-0.10-1	10345444	10363546	9677902	259106	110
LD802-5-0.50-0	10490142	10551129	9950921	228493	110
LD803-5-0.10-1	9538288	9565227	8918853	275070	106
LD803-5-0.25-1	9743314	9815785	8999055	275216	104
LD803-5-0.25-0	9748326	9763289	9274975	256875	111
LD803-5-0.10-0	9696436	9720730	9290951	215221	107
LD804-5-0.10-0	8440937	8464053	8004652	184115	110
LD804-5-0.50-0	8479098	8502467	8075482	168527	102
LD804-5-0.25-0	8444159	8476240	8181376	114853	107
LD804-1-0.25-1	8898841	8965159	8181936	316304	111

Table VII: MWPT: Results for four instances of 120 points.

Par. Setting	Average	Median	Best	Std. Dev.	#Pts
LD1201-5-0.10-1	13941438	14024447	12842149	386460	163
LD1201-5-0.25-0	14119597	14168102	13059200	397667	159
LD1201-5-0.25-1	14364494	14376191	13343235	405153	157
LD1201-5-0.10-0	14078301	14131013	13509895	284575	158
LD1202-5-0.25-0	10092545	10166051	9247582	303468	154
LD1202-5-0.10-0	10143516	10155656	9488995	229203	153
LD1202-5-0.10-1	10109354	10149822	9555352	223387	156
LD1202-5-0.50-0	10213997	10234101	9650748	264040	156
LD1203-5-0.10-1	13024222	13027319	12326883	270411	167
LD1203-5-0.25-1	13197334	13226142	12376353	388390	159
LD1203-1-0.25-1	13958025	14022061	12518659	557306	158
LD1203-5-0.10-0	13366048	13453313	12566722	314829	161
LD1204-5-0.10-0	11211772	11288840	10647886	243568	170
LD1204-5-0.10-1	11333409	11377566	10648542	286156	153
LD1204-5-0.50-0	11328537	11345607	10690162	175935	155
LD1204-5-0.50-1	11676292	11705953	10698947	327513	161

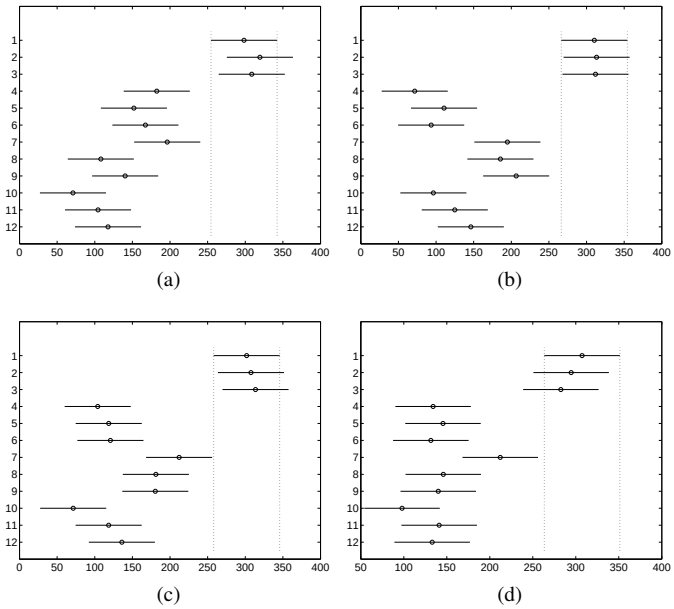

Figure 2: MWPT: Multi-comparison Tukey test: (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.

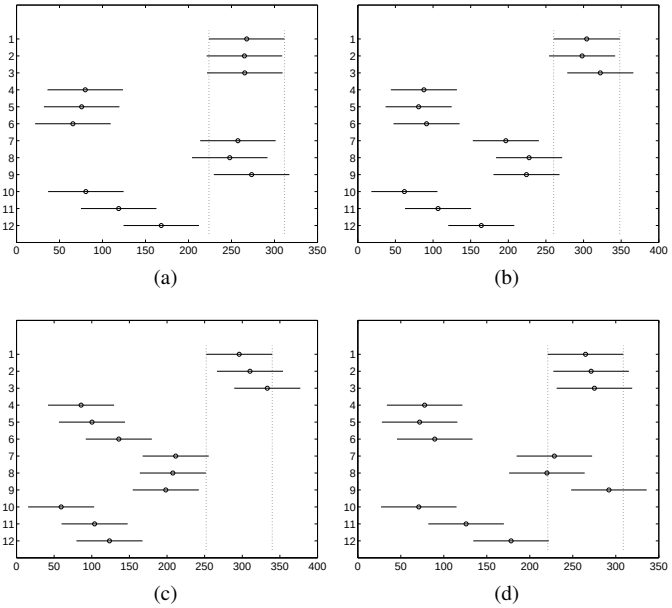

Figure 3: MWPT: Multi-comparison Tukey test: (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.

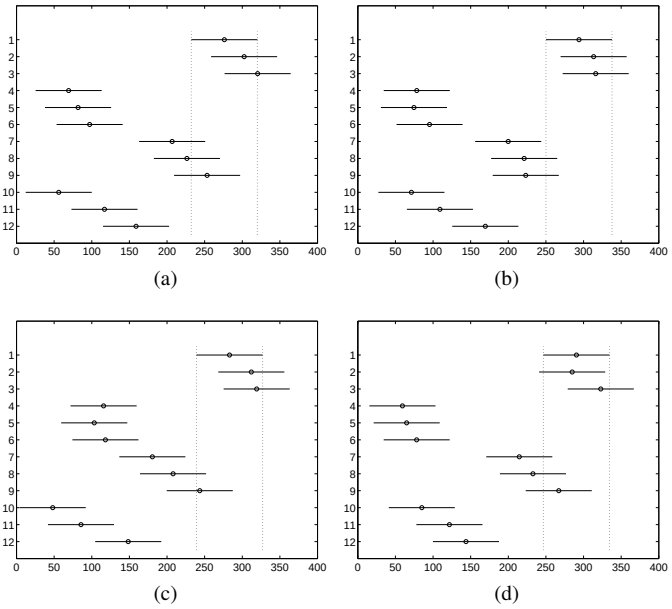

Figure 4: MWPT: Multi-comparison Tukey test: (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.

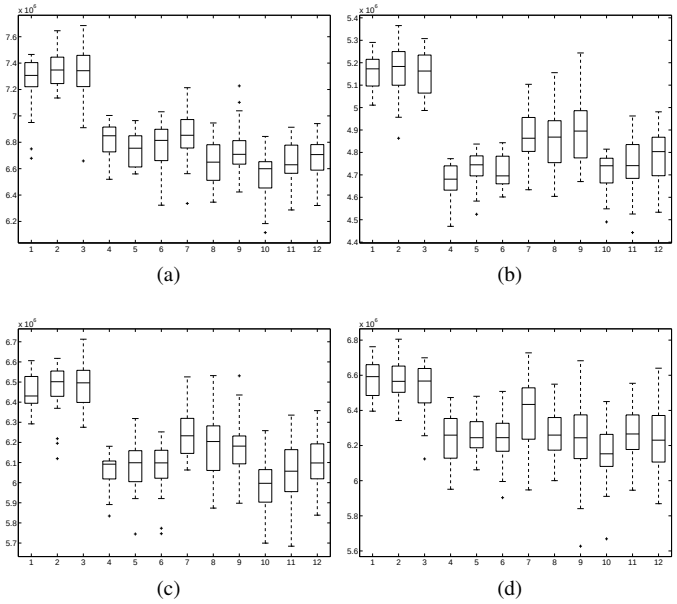

Figure 5: MWPT: Boxplots for (a) LD40-1, (b) LD40-2, (c) LD40-3, and (d) LD40-4.

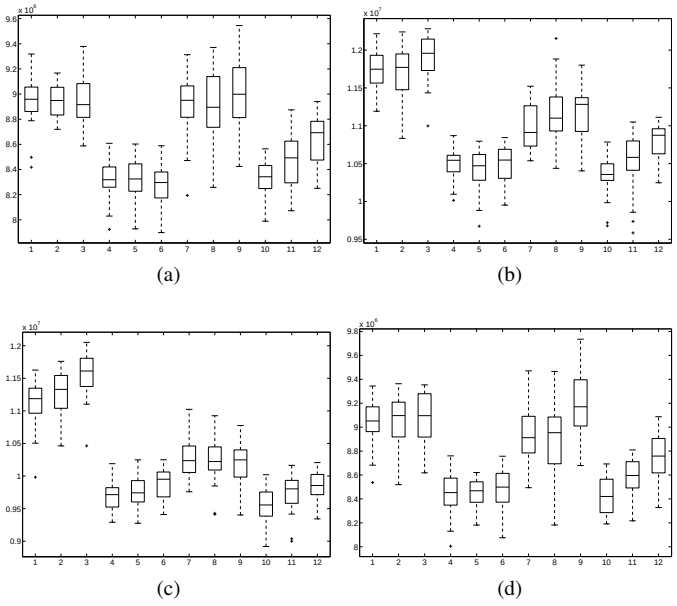

Figure 6: MWPT: Boxplots for (a) LD80-1, (b) LD80-2, (c) LD80-3, and (d) LD80-4.

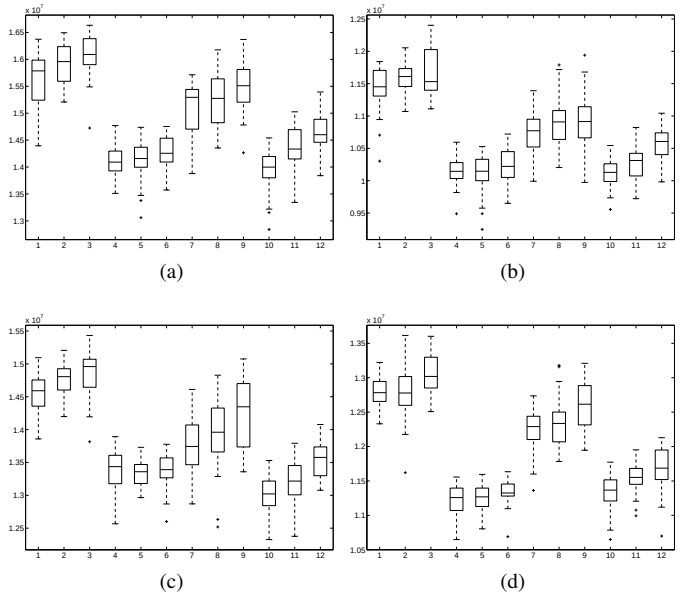

Figure 7: MWPT: Boxplots for (a) LD120-1, (b) LD120-2, (c) LD120-3, and (d) LD120-4.